4 Variables

Included in this chapter:

General Description

Variables are numerical values provided by your analysis software or created within EnSight. Variables can be dependent on part-geometry (for example, the area of a part), and a part's geometry can be dependent on its parent part's variable values (for example, an isosurface).

Variable Types

There are four types of variables: *tensor, vector*, *scalar*, and *constant*. Scalars and vectors can be real or complex. Symmetric tensors are defined by six values, while asymmetric tensors are defined by nine values. Vectors, such as displacement and velocity, have three values (the components of the vector) if real, or six values if complex. Scalars, such as temperature or pressure, have a single value if real, or two values if complex. Constants have a single value for the model, such as analysis time or volume. All four types can change over time for transient models.

Activation

Before using a variable, it must be loaded by EnSight, a process called activation. EnSight normally activates variables as they are needed. Section 4.1 describes how to select, activate, and deactivate variables to make efficient use of your system memory.
(see Section 4.1, Variable Selection and Activation)

Creation

In addition to using the variables given by your analysis software, EnSight can create additional variables based on any existing variables and geometric properties of parts. EnSight provides more than fifty functions (and more are being added for the next minor release) to make this process simpler.(see Section 4.3, Variable Creation)

Color Palettes

Very often you will wish to color a part according to the values of a variable. EnSight associates colors to values using a *color palette*. You have control over the number of value-levels of the palette and the type of scale, as well as control over colors and method of color gradation. You also use function palettes to specify a set of levels for a variable, such as when creating contours.
(see Section 4.2, Variable Summary & Palette)

Queries

You can make numerical queries about variables and geometric characteristics of Server-based parts. These queries can be at points, nodes, elements, parts, along lines, and along 1D parts. If you have transient data, you can query at one time step or over a range of time steps, looking at actual variable values or a Fast Fourier Transform (FFT) of the values. (see Section 6.3, Query Menu Functions)

Plotting

Once you have queried a variable, you can plot the result.

(see Section 8.5, Plot Mode)

From More than One Case

Variables can come from more than one case. If more than one case has a variable with the same name, this will be treated as one variable. If a variable is applicable to one case but not another, it will not be applied to the non-applicable case(s).

Parts

When variables are activated or created, all parts except Particle Trace parts are updated to reflect the new variable state. Particle Trace parts will always show variables which are activated after the part's creation as zero values.

Location

Variables can be defined at the vertices, at the element centers, or undefined.

User Defined Math Functions

Users can write external variable calculator functions called User Defined Math Functions (UDMF) that can be dynamically loaded by EnSight. These functions appear in EnSight's calculator in the general function list and can be used just as any other calculator function to derive new variables.

Several examples of UDMFs can be found in the directory `$CEI_HOME/ensight74/user_defined_src/math_functions/`. Please see these examples if you wish to create your own UDMFs.

When the EnSight server starts it will look in the following subdirectories for UDMF dynamic shared libraries:

```
./libudmf-devel.so (.sl) (.dll)
$ENSIGHT7_UDMF/libudmf-*.so (.sl) (.dll)
$CEI_HOME/ensight74/machines/$ENSIGHT7_ARCH/lib_udmf/libudmf-*.so (.sl)
(.dll)
```

Depending on the server platform, the dynamic shared library must have the correct suffix for that platform (e.g. `.so, .sl, .dll`).

Currently, when a UDMF is used in the EnSight calculator, it in invoked for each node in the specified part(s) if all the variables operated on for the specified part(s) are node centered. If all of the variables are element centered, then the UDMF is invoked for each element in the part(s). If the variables are a mix of node and element centered values, then the node centered values are automatically converted to element centered values and then the UDMF is invoked for each element using element centered variables.

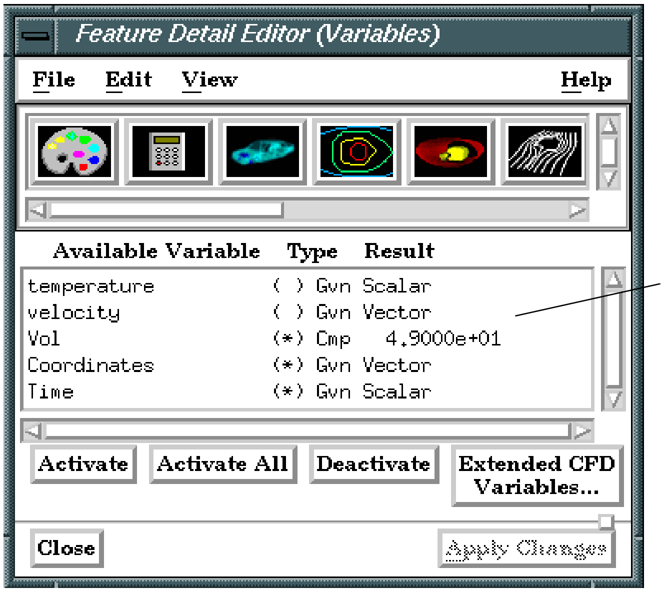
Arguments and the return type for the UDMF can be either scalar or vector EnSight variables or constants. At this time, only variable quantities and constants can be passed into UDMFs. There is no mechanism for passing in either part geometry, neighboring variables, or other information.

4.1 Variable Selection and Activation

All available variables, both those read in and those created within EnSight, are shown in the Feature Detail Editor (Variables), whether they have been activated or not. In addition, a variable list is included in each function requiring a variable. In this case, only the appropriate variable types are shown.

Feature Detail Editor (Variables)

Double clicking on the Color Icon in the Feature Icon Bar opens the Feature Detail Editor (Variables).



Available Variable	Type	Result
temperature	( ) Gvn Scalar	
velocity	( ) Gvn Vector	
Vol	(*) Cmp	4.9000e+01
Coordinates	(*) Gvn Vector	
Time	(*) Gvn Scalar	

Feature Detail Editor Variables List

Figure 4-1
Feature Detail Editor (Variables)

Feature Detail Editor Variables List

This list shows all variables currently available, both those read from data and those you have created within EnSight. Each row provides information about a variable.

Available Variable

The description or name of the variable.

( ) or (*)

Activation status. An asterisk indicates that the variable has been activated.

Type

Type of the variable:

Gvn Scalar:	real scalars read from the dataset (Given).
Cmp Scalar:	real scalars created within EnSight (Computed).
Gvn Complex Scalar:	complex scalars read from the dataset (Given).
Cmp Complex Scalar:	complex scalars created within EnSight (Computed).
Gvn Vector:	real vectors read from the dataset (Given).
Cmp Vector:	complex vectors created within EnSight (Computed).
Gvn Complex Vector:	complex vectors read from the dataset (Given).
Cmp Complex Vector:	complex vectors created within EnSight (Computed).
Gvn Tensor:	real tensors read from the dataset (Given).
Cmp Tensor:	real tensors created within EnSight (Computed).
Gvn #:	constants read from the dataset (Given).
Cmp #:	constants created within EnSight (Computed).

Result

Current value of a constant variable (is blank for other types of variables). Changing the current solution time will update the value in this column to the value for the new time.

Activate

Clicking this button activates the variable(s) selected in the Feature Detail Editor Variables List. Activation of a variable loads its values into the memory of the EnSight Server host system. The EnSight Server then passes the necessary data to the Client. One way you can control EnSight's memory usage is to only activate the variables you want to use. Once activated, a variable becomes available in the Main Variables List and, as is described in Section 4.2, EnSight creates a default color palette for the variable.

Activate All

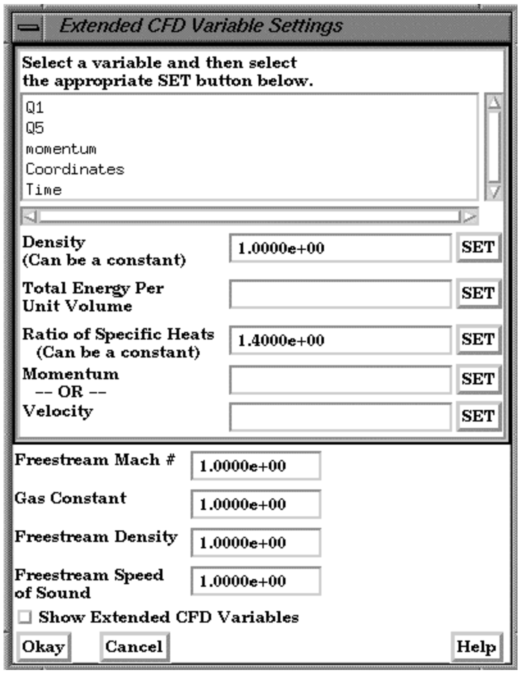
Clicking this button activates all variables listed in the Feature Detail Editor Variables List, regardless of which are selected.

Deactivate

Clicking this button deactivates the variable(s) selected in the Feature Detail Editor Variables List. Deactivating a variable frees up some memory on both the Client and the Server. You can activate and deactivate variables as often as you like. For example, you could activate one variable to color a part, deactivate that variable, then activate a different variable to re-color the part. Of course, if you have enough memory and a small enough model, you can simply activate all the variables and leave them activated.

Extended CFD Variables...

Opens the Extended CFD Variable Settings dialog. If your data defines variables or constants for density, Total Energy per unit volume, and momentum (or velocity), it is possible to show new variables defined by these basic variables in the Main Variables List of the GUI by utilizing the capabilities of this dialog. (See Preferences... in Section 6.2, Edit Menu Functions).



Extended CFD Variable Settings

Select a variable and then select the appropriate SET button below.

Q1
Q5
momentum
Coordinates
Time

Density (Can be a constant): 1.0000e+00

Total Energy Per Unit Volume:

Ratio of Specific Heats (Can be a constant): 1.4000e+00

Momentum -- OR -- Velocity:

Freestream Mach #: 1.0000e+00

Gas Constant: 1.0000e+00

Freestream Density: 1.0000e+00

Freestream Speed of Sound: 1.0000e+00

□ Show Extended CFD Variables

Figure 4-2
Extended CFD Variable Settings Dialog

WARNING If you deactivate a created variable or any of the variables used to define it, both the values and the definition of the created variable are deleted. If you deactivate a variable used to create a part's geometry, the part will be deleted. If you deactivate a variable who's color palette has been used to color a part, the part's appearance will change.

(see How To Activate Variables)

4.2 Variable Summary & Palette

You can visualize information about a model by representing variable values with colors, often called fringes. Fringes are an extremely effective way to visualize variable variations and levels. A variable color palette associates (or maps) variable values to colors. Palettes are also used in the creation of contours. The number of contour levels is based on the number of palette color levels, and the contour values are based on the palette level values.

EnSight uses a variable's color palette to convert numbers to colors, while you, the viewer, use them in the opposite manner—to associate a visible color with a number. If you wish, EnSight can display a color-value legend in the Main View window.

Default Palettes

At least one color palette—the Coordinate color palette—always exists, even if your model has no variables. In addition, EnSight creates a color palette for each real scalar and vector variable that you activate, giving the color palette the same name as the variable. If the variable is a vector variable, the default color palette uses the vector's magnitude. Tensor variables have no palette.

Default color palettes have five color levels. Ranging from low to high, the colors are blue, cyan, green, yellow, and red (the spectral order). The numerical values mapped to these five levels are determined by first finding the value-range for the variable at the current time step when the variable is activated. The value for the lowest level is set to the minimum value. The value for the highest level is set to the maximum value. The three middle levels are spaced evenly between the lowest and highest values. For datasets with only one time step, the scheme just described works well because the variable's value range is not changing over time.   However, if you have transient data, the range could vary widely at different times and since the default was based on one time step, it may not be appropriate for other time steps. EnSight can show you a histogram of the variable values over time to assist you in setting a palette for transient cases.

Value Levels

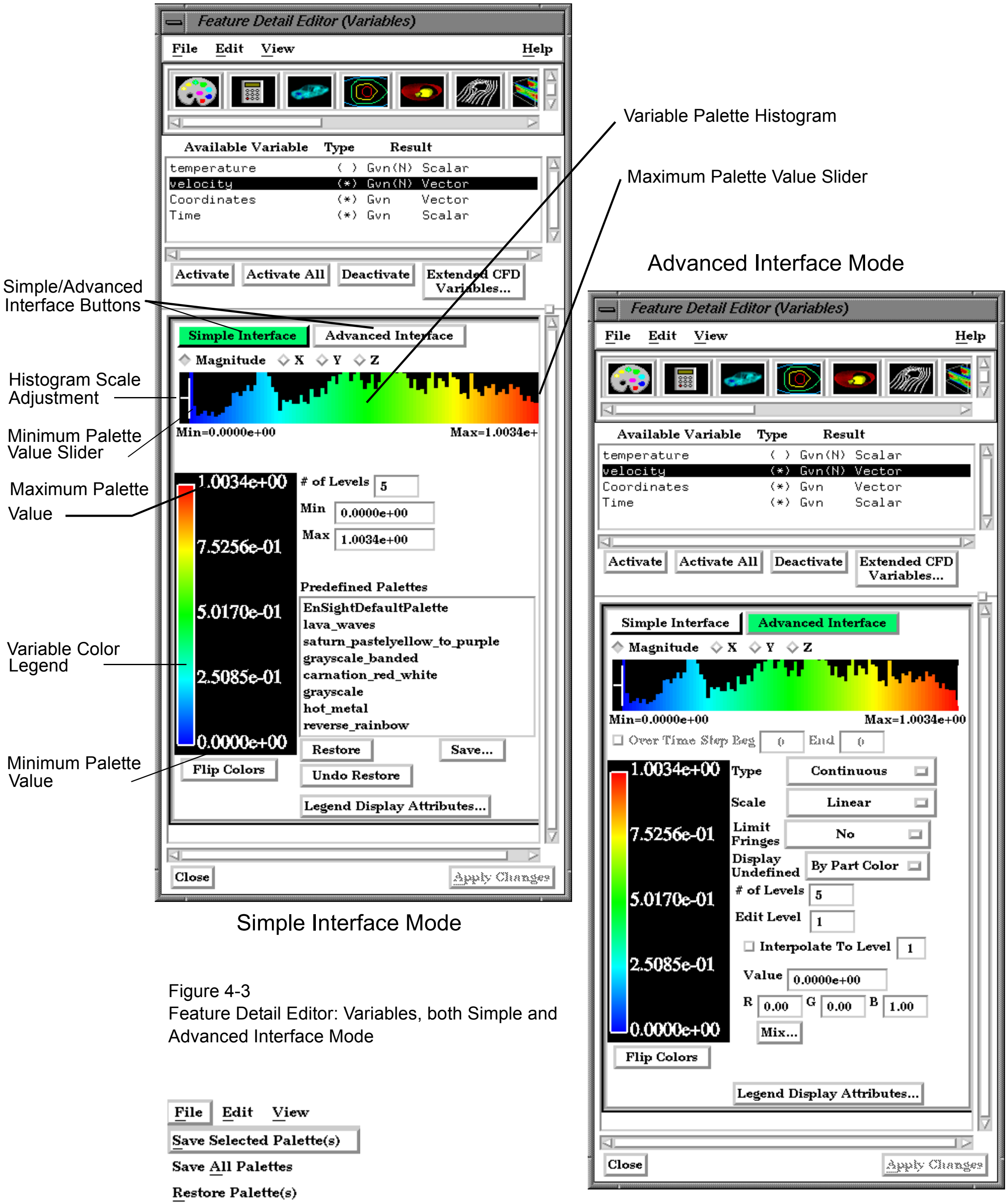
A color palette can have up to 21 levels at which the variable value is specified. Each color palette level's value must be between the value at the adjoining levels, with higher levels having higher variable-values. Between levels, you select whether the scale is linear (the default), quadratic (2^x), or logarithmic ($\log_{10}$). Also, you can have EnSight use one of these scales to automatically assign values to a range of levels.

Sometimes you may wish to only visualize areas whose palette-variable values are in a limited range. You can choose to visualize other areas with a different, uniform color, or to make those areas invisible.

Management

The Feature Detail Editor (Variables) enables you to manage your color palettes. You can copy, save to a file, and restore from a file existing palettes

Clicking the Variable Summary and Palette turndown button opens that dialog within the Feature Detail Editor (Variables) dialog.



Available Variable	Type	Result
temperature	( ) Gvn(N)	Scalar
velocity	(*) Gvn(N)	Vector
Coordinates	(*) Gvn	Vector
Time	(*) Gvn	Scalar

◆ Magnitude ◇ X ◇ Y ◇ Z

Min=0.0000e+00 Max=1.0034e+00

1.0034e+00
7.5256e-01
5.0170e-01
2.5085e-01
0.0000e+00

Simple Interface:

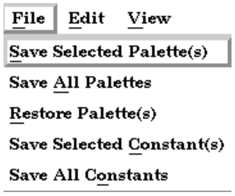
# of Levels 5
Min 0.0000e+00
Max 1.0034e+00

Predefined Palettes
EnSightDefaultPalette
lava_waves
saturn_pastelyellow_to_purple
grayscale_banded
carnation_red_white
grayscale
hot_metal
reverse_rainbow

Advanced Interface:

□ Over Time Step Beg 0 End 0
Type Continuous
Scale Linear
Limit Fringes No
Display Undefined By Part Color
# of Levels 5
Edit Level 1
□ Interpolate To Level 1
Value 0.0000e+00
R 0.00 G 0.00 B 1.00

Figure 4-3
Feature Detail Editor: Variables, both Simple and Advanced Interface Mode



File Menu Clicking this button opens a pull-down menu with the following options:

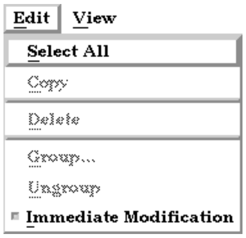
Save Selected Palette(s) Opens the file selection dialog for the specification of a filename in which to save the selected color palette(s).

Save All Palettes… Opens the file selection dialog for the specification of a filename in which to save all color palette(s).

Restore Palette(s) Opens the file selection dialog for the specification of a filename from which to restore previously saved color palettes.

Save Selected Constant(s) Opens the file selection dialog for the specification of a filename in which to save the selected constant values.

Save All Constant(s) Opens the file selection dialog for the specification of a filename in which to save all constant values.



Edit Menu Clicking this button opens a pulldown menu with the following choice:

Select All Clicking this selects all variables in the Feature Detail Editor Available Variables List.

Immediate Modification Toggle Default is On. While on, any modification made in the Editor is immediately implemented by EnSight. For large problems, this may be impractical. In such instances, click this toggle off, make all desired modifications, and then implement then all at once by clicking the Apply Changes button at the bottom of the Editor dialog.

Simple/Advanced Interface Buttons which allow the user to choose between a simple or advanced mode for this dialog. The advanced interface is shown in the figure. The simple interface is a small subset of the advanced.

Variable Palette Histogram This histogram shows the relative number of nodes at which the value of the selected variable is within the range represented by a particular color band. The two vertical white slider bars are used to interactively set the minimum and maximum variable values to be used in the variable's color palette and these will show up in the Legend both within the turndown area and within the Graphics Window. The small horizontal white line on the left hand side can be used to interactively adjust the vertical scale of the histogram.

Over Time Step Toggle & Beg, End Fields Toggles on/off the automatic assignment of values to palette levels using the palette-variable's value range over multiple time steps which are specified in the Beg and End fields to the right of the toggle. This function is only available when you are using transient data. All other attributes of the color palette (including the number of levels, colors, type, etc.) are not changed.

Magnitude, X,Y,Z, Toggles For vector variables, this controls which histogram and color palette will be displayed and edited. By default, the vector magnitude is used, however, the X, Y, and Z components of the vector are also available.

Type

This button opens a pop-up menu for the selection of the desired type of color gradation. Both the legend in the turn-down area and the legend in the Graphics Window (if visible) are affected. Options are:

Continuous displays graduated color variation across or along each element interpolating the color across each element based on the value of the variable at the nodes. If the variable tied to the palette is defined at the element centers it will be averaged to the nodes for display.

Banded displays discrete color values for each value range, but interpolates the location demarcation line within an element.

Constant displays each element with one color for the entire element rather than interpolating the color across the element using values at the nodes. The color of the first node encountered is used.

Scale

This button opens a pop-up dialog for the selection of the desired type of scale for the value-separation of levels and color gradation. The options are:

Linear scale divisions, where the value-separation of levels is uniform and values map linearly to the colors.

Quadratic scale divisions, where the value-separations of levels are not equal, but instead are based on the second order of the variable (value2). Level-values always increasing upwards. For example, for five levels with a low-level value of 0 and a high-level value of 16, the linear scale would be 0, 4, 8, 12, 16 while the quadratic scale would be 0, 1, 4, 9, 16.

Logarithmic scale divisions, where the value-separations of levels are not equal, but instead are based on the base–10 logarithm of the variable value (log10). Level-values always increasing upwards. For example, for five levels with a low-level value of 1 and a high-level value of 10000, the linear scale would be 1, 2500, 5000, 7500, 10000 while the logarithmic scale would be 1, 10, 100, 1000, 10000.

Limit Fringes

This button allows you to select how you wish to display elements with node values above and below the range of the palette scale values. This option only works for hidden surface mode. Options are:

No limit on values. Values above and below are colored with color of the corresponding end of the range (no interpolation).

By Model Color option colors values outside the function range with the current part- color (the color of the part when its Color By Palette attribute is None).

By Invisible option does not display elements whose node values are all above or below the value-range of the palette.

Display Undefined

If the variable is not defined, the element cannot be colored according to the color palette. In this case, the element will be colored by the Part Color, or the element will become invisible.

# of Levels

This field specifies the number of value-levels for the variable color palette, which are shown beside the Legend color bar. The number of levels is independent of the Type and Scale, and can range from 2 to 21 with the default being 5.

Min

For the Simple Interface, this field is used to specify the variable value for the bottom level.

Max

For the Simple Interface, this field is used to specify the variable value for the top level.

Edit Level

Selection of the level you wish to edit, selected with stepper buttons, by entering a value in the field, or by clicking the mouse pointer on the desired level in the Variable Color Legend area. Levels start at 1 and count up from lower end. You can change the variable-value and color assigned to any level. Also, you can have EnSight interpolate value-levels and colors over a range of levels.

Interpolate to Level Toggle and Field

If this option is toggled-on while you are specifying a value (or color), the value (or color) of EnSight adjusts the values (or colors) of intermediate levels between the current level and the specified Interpolate To Level according the specified Scale type.

Value

This field specifies the variable value for the current palette level.

R G B Fields These fields are used to specify the color to use for the current palette level.

Mix... Clicking this button opens the Color Selector dialog which provides an alternative to the RGB fields for the specification of the color to use for the current palette level. (see Section 7.1, Color)

Predefined Palettes For the Simple Interface only, shows a list of all predefined color palettes.

Restore For the Simple Interface only. Restores the palette selected in the Predefined Palettes list.

Save... For the Simple Interface only. Will bring up a file dialog to allow saving of the currently defined color palette.

Undo Restore For the Simple Interface only. Will set the color palette definition back to what existed before the previous Restore.

Flip Colors Reverses colors in the palette.

Legend Display Attributes ... Clicking this button opens a pop-up message which reminds you that additional options for the modification of Legend display attributes may be found in the Annot Mode Icon Bar. (See How To Create Color Legends, How To Edit Color Palettes)

4.3 Variable Creation

You can create additional variables based on existing data. Typical mathematical operations, as well as many special built-in functions, enable you to produce simple or complex equations for new variables. Some built-in functions enable you to use values based on the geometric characteristics of parts. Created variables are available for any process, just like given variables. If you have transient data, a time change will recompute the created variable values.

Often an analysis program produces a set of basic results from which other results can be derived. For example, if a computational fluid dynamics analysis gives you density, momentum and total energy, you can derive pressure, velocity, temperature, mach number, etc. EnSight provides many of these common functions for you, or you can enter the equation(s) and build your own.

As another example, suppose you would like to normalize a given scalar or vector variable according to its maximum value, or according to the value at a particular node. Variable creation enables you to easily accomplish such a task. The more familiar you become with this feature, the more uses you will discover.

EnSight allows variables to be defined at vertices (nodes) or element centers. If a new variable is created from a combination of nodal and element based variables, such a new variable will always be element based.

Building Expressions

The Feature Detail Editor (Variables) dialog Variable Creation turn-down section provides function selection lists, calculator buttons, and feedback guidance to aid you in building the working expression (or equation) for a new variable. You can use three types of values in an expression: constants, scalars, and vectors.

Constants

A *constant* in a variable expression can be a…	for example…
• number	3.56
• constant variable from the Active Variables list	Analysis_Time
• scalar variable at a particular node/element (component and node/element number in brackets)	temperature[25]
• vector variable component at a particular node /element (component and node/element number in brackets)	velocity[Z][25]
• coordinate component at a particular node/element (component and node/element number in brackets)	coordinate[X][25]
• any of the previous three at a particular time step	temperature{15}[25]
• (time step in braces right after the variable name) coordinate{15}[X][25]	velocity{15}[Z][25]
• Math function	COS(1.5708)
• General function that produces a constant	AREA(plist)

Scalars

A *scalar* in a variable expression can be a…	for example…
• Scalar variable from the Active Variables list	pressure
• vector variable component (component in brackets)	velocity[Z]
• coordinate component (component in brackets)	coordinate[Y]
• any of the previous three at a particular time step (time step in braces right after the variable name) coordinate{29}[Y]	pressure{29} velocity{29}[Z]
• General function that produces a scalar	Divergence(plist,velocity)

Vectors	A *vector* in a variable expression can be a…	for example…
	• vector variable from the Active Variables list	velocity
	• coordinate name from the Active Variables list	coordinate

	• any of the previous two at a particular time step (time step in braces right after the variable name)	velocity{9}
		coordinate{9}

	• General function that produces a vector	Vorticity(plist,velocity)

Examples of Expressions and How To Build Them

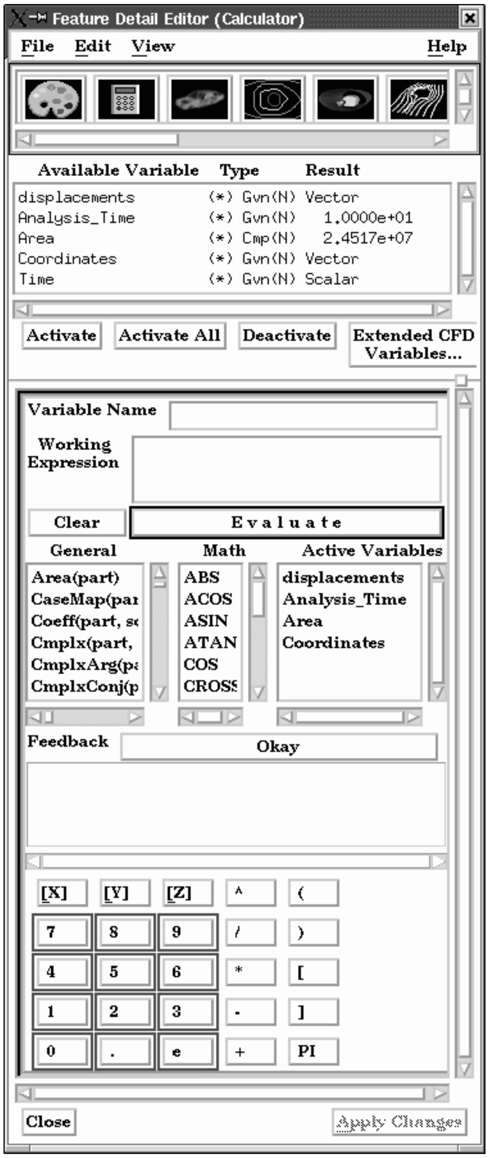
The following are some example variable expressions, and how they can be built. These examples assume Analysis_Time, pressure, density, and velocity are all given variables.

Expression	*Discussion and How To Build It*
-13.5/3.5	A true constant since it does not change over time. To build it, type on the keyboard or click on the Variable Creation dialog calculator buttons `-13.5/3.5`
Analysis_Time/60.0	A simple example of modifying a given constant variable. If Analysis_Time is in seconds, this expression would give you the value in minutes. To build it, select *Analysis_Time* from the Active variable list and then type or click `/60.0`.
velocity*density	This expression is momentum, which is a vector. To build it, select *velocity* from the Active Variables list, type or click *, then select *density* from the Active Variable list.
SQRT(pressure[73] * 2.5)+ velocity[X][73]	This says, take the pressure at node (or element if pressure is an element center based variable) number 73, multiply it by 2.5, take the square root of the product, and then add to that the x-component of velocity at node (or element) number 73. To build it, select *SQRT* from the Math function list, select *pressure* from the Active Variables list, type `[73]*2.5)+`, select *velocity* from the Active Variable list, then type `[X][73].`
pressure{19}	This is a scalar, the value of pressure at time step 19. It does not change with time. To build it, select pressure from the Active Variables list, then type `{19}`.
MAX(plist,pressure)	MAX is one of the built-in General functions. This expression calculates the maximum pressure value for all the nodes of the selected parts. To build it, type or click (, select *MAX* from the General function list and follow the interactive instructions that appear in the Feedback area of this dialog (in this case, to select the parts, click Okay, and select *pressure* from the Active Variable list).
(pressure /pressure_max)^2	This scalar is essentially the normalized pressure, squared. To build it, first build the preceding MAX(plist,pressure) expression and name it "pressure_max". Then to build this expression, select *pressure* from the Active Variables list, type or click `/`, select *pressure_max* from the Active Variables list, then type or click `)^2`.

Notice in the last example how a complex equation can be broken down into several smaller expressions, just as in a programming language. Simply assign a variable to the intermediate expression and include that variable name in later expressions. In fact, this is a necessary step in many cases since EnSight can compute only one variable at a time. It is not valid to use a variable name for the new variable which is used in the expression. For example, it is not valid to create an expression such as:

temperature = temperature + 100.

Clicking the Calculator Icon opens the Feature Detail Editor (Calculator) dialog.



```
displacements        (*) Gvn(N) Vector
Analysis_Time        (*) Gvn(N)   1.0000e+01
Area                 (*) Cmp(N)   2.4517e+07
Coordinates          (*) Gvn(N) Vector
Time                 (*) Gvn(N) Scalar
```

General: Area(part), CaseMap(par, Coeff(part, s, Cmplx(part,, CmplxArg(p, CmplxConj(p

Math: ABS, ACOS, ASIN, ATAN, COS, CROSS

Active Variables: displacements, Analysis_Time, Area, Coordinates

Figure 4-4
Feature Detail Editor (Calculator) dialog

Variable Name This field is used to specify the name for the variable being created. Built-in general functions will provide a default, but they can be modified here. Variable names must not start with a numeric digit and must not contain any of the following reserved characters:

(	[	{	+	@	!	*	$
)	]	}	–	space	#	^	/

Working Expression

The expression or equation for the new variable is presented in this area. Interaction with the expression takes place here, either directly by typing in values and variable names, etc., or indirectly by selecting built-in functions and clicking calculator buttons.

Clear

Clicking this button clears the Variable name field, Working Expression area, Feedback area, and deselects any built-in function.

Evaluate

Clicking this button produces the new variable defined in the working expression area. Until you click this button, nothing is really created. The selection commands specify to which parts the new variable should be applied.

General

Scroll this list of built-in functions provided for your convenience. Click on a function to insert it into your Working Expression. For some functions, the Feedback Window provides interactive instructions.

Area

Area (any part(s))

Computes a constant variable whose value is the area of the selected parts. If a part is composed of 3D elements, the area is of the border representation of the part. The area of 1D elements is zero.

Case Map

CaseMap(2D or 3D part(s), case to map from, scalar or vector)

Finds the specified scalar or vector variable values for the specified part(s) from the indicated case.

Coefficient

Coeff (any 1D or 2D part(s), scalar, component)

Computes a constant variable whose value is a coefficient Cx , Cy, or Cz such that

$$C_x = \underset{S}{f n_x dS} \qquad C_y = \underset{S}{f n_y dS} \qquad C_z = \underset{S}{f n_z dS}$$

where:

f = any scalar variable
S = 1D or 2D domain
n_x = x component of normal
n_y = y component of normal
n_z = z component of normal

Specify [X], [Y], or [Z] to get the corresponding coefficient.

Note: Normal for a 1D part will be parallel to the plane of the plane tool.

Complex

Cmplx(any part(s), real portion, complex portion, frequency)

Creates a complex scalar or vector from two scalar or vector variables. The frequency is optional and is used only for reference.

Z = A + Bi

Complex Argument

CmplxArg(any part(s), complex scalar or vector)

Computes the Argument of a complex scalar or vector. The resulting scalar is given in degrees and will be in the range -180 and 180 degrees.

Arg = atan(Vi/Vr)

Complex Conjugate

CmplxConj(any part(s), complex scalar or vector)

Computes the Conjugate of a complex scalar of vector. Returns a complex scalar or vector where:

Nr = Vr
Ni = -Vi

Complex Imaginary *ComplexImaginary*(any part(s), complex scalar or vector)

Extracts imaginary portion of a complex scalar or vector into a real scalar or vector.

N = Vi

Complex Modulus *ComplexModulus*(any part(s), complex scalar or vector)

Returns a real scalar/vector which is the modulus of the given scalar/vector

N = SQRT(Vr*Vr + Vi*Vi)

Complex Transient Response *CmplxTransientResponse*(any part(s), complex scalar or vector, angle (degrees))

Returns a real scalar or vector which is the real transient response:

Re(Vt) = Re(Vc)Cos(phi) - Im(Vc)Sin(phi)

which is a function of the transient phase angle “phi” defined by:
phi = 2 Pi f t
where
t = the harmonic response time parameter
f = frequency of the complex variable “Vc”

and the complex field “Vc”, defined as:
Vc = Vc(x,y,z) = Re(Vc) + i Im(Vc)
where
Vc = the complex variable field
Re(Vc) = the Real portion of Vc
Im(Vc) = the imaginary portion of Vc
i = Sqrt(-1)

Note, the transient complex function, was a composition of Vc and Euler’srelation, namely:

Vt = Vt(x,y,z,t) = Re(Vt) + i Im(Vt) = Vc * e^(i phi)

where:
e^(i phi) = Cos(phi) + i Sin(phi)

The real portion Re(Vt), is as designated above:

Note: this function is only good for harmonic variations, thus fields with a defined frequency!

Ensight allows phi to vary between 0 and 360 degrees.

Complex Real *ComplexReal*(any part(s), complex scalar or vector)

Extracts the real portion of a complex scalar or vector into a real scalar or vector.

N = Vr

Curl *Curl* (any part(s), vector)

Computes a vector variable which is the curl of the input vector

$$Curl_f = \bar{\nabla} \times \bar{f} = \left(\frac{\partial f_3}{\partial y} - \frac{\partial f_2}{\partial z}\right)\hat{i} + \left(\frac{\partial f_1}{\partial z} - \frac{\partial f_3}{\partial x}\right)\hat{j} + \left(\frac{\partial f_2}{\partial x} - \frac{\partial f_1}{\partial y}\right)\hat{k}$$

Density Density(any part(s), pressure, temperature, gas constant).

Computes a scalar variable which is the density ρ , defined as:

$$\rho = \frac{p}{TR}$$

where: p = pressure
T = temperature
R = gas constant

Normalized **Density**

DensityNorm (any part(s), density, freestream density)

Computes a scalar variable which is the Normalized Density ρ_n defined as:

$$\rho_n = \rho / \rho_i$$

where: ρ = density

ρ_i = freestream density

Log of Normalized **Density**

DensityLogNorm (any part(s), density, freestream density)

Computes a scalar variable which is the natural log of Normalized Density defined as:

$$ln \rho_n = ln(\rho / \rho_i)$$

where: ρ = density

ρ_i = freestream density

Stagnation **Density**

DensityStag (any part(s), density, total energy, velocity, ratio of specific heats)

Computes a scalar variable which is the Stagnation Density ρ_o defined as:

$$\rho_o = \rho\left(1 + \left(\frac{\gamma - 1}{2}\right)M^2\right)^{(1/(\gamma - 1))}$$

where: ρ = density

γ = ratio of specific heats

M = mach number

Normalized Stagnation **Density**

DensityNormStag (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, freestream velocity magnitude)

Computes a scalar variable which is the Normalized Stagnation Density ρ_{on} defined as:

$$\rho_{on} = \rho_o / \rho_{oi}$$

where: ρ_o = stagnation density

where: ρ_{oi} = freestream stagnation density

Divergence

Div (any 2D or 3D part(s), vector)

Computes a scalar variable whose value is the divergence defined as:

$$Div = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}$$

where u,v,w = velocity components in x,y,z directions.

Element to Node

ElemToNode (any part(s), element based variable).

Averages an element based variable to produce a node based variable.

Energy:

Total **Energy** *EnergyT* (any part(s), density, pressure, velocity, ratio of specific heats).
Computes a scalar variable of total energy per unit volume

$$e = \rho\left(e_i + \frac{V^2}{2}\right)$$ Total Energy

$$e_i = e_0 - \frac{V^2}{2}$$ Internal Energy

$$e_0 = \frac{e}{\rho}$$ Stagnation Energy

where:

$\rho = density$

$V = Velocity$

Or based on gamma, pressure and velocity:

$$e = \frac{p}{(\gamma - 1)} + \rho\frac{V^2}{2}$$

Kinetic **Energy** *KinEn* (any part(s), velocity, density)
Computes a scalar variable whose value is the kinetic energy E_k defined as:

$$E_k = \frac{1}{2}\rho V^2$$

where ρ = density (variable or constant)
V = Velocity variable

Enthalpy *Enthalpy* (any part(s), density, total energy, velocity, ratio of specific heats)
Computes a scalar variable which is Enthalpy h defined as:

$$h = \gamma\left(\frac{E}{\rho} - \frac{V^2}{2}\right)$$

where: E = total energy per unit volume
ρ = density
V = velocity magnitude
γ = ratio of specific heats

Normalized **Enthalpy** *EnthalpyNorm* (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound)
Computes a scalar variable which is Normalized Enthalpy h_n defined as:

$$h_n = h/h_i$$

where: h = enthalpy
h_i = freestream enthalpy

Stagnation **Enthalpy** *EnthalpyStag* (any part(s), density, total energy, velocity, ratio of specific heats) Computes a scalar variable which is Stagnation Enthalpy h_o defined as:

$$h_o = h + \frac{V^2}{2}$$

where: h = enthalpy
V = velocity magnitude

Normalized Stagnation Enthalpy

EnthalpyNormStag (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, freestream velocity magnitude)

Computes a scalar variable which is Normalized Stagnation Enthalpy h_{on} defined as:

$$h_{on} = h_o / h_{oi}$$

where: h_o = stagnation enthalpy

h_{oi} = freestream stagnation enthalpy

Entropy

Entropy (any part(s), density, total energy, velocity, ratio of specific heats, gas constant, freestream density, freestream speed of sound)

Computes a scalar variable which is Entropy s defined as:

$$s = ln\left(\frac{\frac{p}{p_i}}{\left(\frac{\rho}{\rho_i}\right)^{\gamma}}\right)\left(\frac{R}{\gamma - 1}\right)$$

where: R = gas constant

ρ = density

ρ_i = freestream density

p = pressure

p_i = freestream pressure = $(\rho_i c_i^2)/\gamma$

c_i = velocity magnitude

γ = ratio of specific heats

Flow

Flow (any 1D or 2D part(s), velocity).

Computes a constant variable whose value is the flow Q_c defined as:

$$Q_c = \underset{S}{} V_n dS$$

where V_n = Velocity value normal to the surface

S = 1D or 2D domain

Note: Normal for a 1D part will be parallel to the plane of the plane tool

Flow Rate

FlowRate (any 1D or 2D part(s), velocity).

Computes a scalar variable Q defined as:

$$Q = V \cdot N$$

where V = Velocity

N = Surface Normal

Fluid Shear *FluidShear*(2D part(s), velocity gradient, viscosity)

Computes a scalar variable tau whose value is defined as:

$$\text{tau} = \mu\frac{\partial V}{\partial n} \qquad \text{where tau = shear stress}$$

μ = dynamic viscosity

$\frac{\partial V}{\partial n}$ = Velocity gradient in direction of surface normal

Hints: To compute fluid shear stress:

1. Use gradient function on velocity to obtain “Velocity Grad” variable in the 3D part(s) of interest.
2. Use clip option (through the 3D part(s) used in 1.) to obtain a surfaceon which you wish to see the fluid shear stress.
3. Compute Fluid Shear variable (on the 2D clip surface of 2.)

Fluid Shear Stress Max *FluidShearMax* (3D part(s), velocity, density, turbulent kinetic energy, turbulent dissipation, laminar viscosity)

Computes a scalar variable Σ defined as:

$$\Sigma = F/A = (u_t + u_l)E \qquad \text{where } F = \textit{force}$$

A = *unit area*

u_t = *turbulent (eddy) viscosity*

u_l = *laminar viscosity (treated as a constant)*

E = *local strain*

The turbulent viscosity u_t is defined as:

$$u_t = \frac{\rho 0.09\kappa^2}{\varepsilon} \qquad \text{where } \rho = \textit{density}$$

κ = *turbulent kinetic energy*

ε = *turbulent dissipation*

A measure of local strain E (i.e. local elongation in 3 directions) is given by

$$E = \sqrt{(2tr(D\cdot D))} \qquad \text{where}$$

$$2tr(D\cdot D) = 2((d_{11})^2 + (d_{22})^2 + (d_{33})^2) + ((d_{12})^2 + (d_{13})^2 + (d_{23})^2)$$

given the *Euclidean norm* defined by

$$tr(D\cdot D) = (d_{11})^2 + (d_{22})^2 + (d_{33})^2 + \frac{1}{2}((d_{12})^2 + (d_{13})^2 + (d_{23})^2) \quad ;$$

and the rate of deformation tensor *dij* defined by

$$D = [d_{ij}] = \frac{1}{2}\begin{bmatrix} 2d_{11} & d_{12} & d_{13} \\ d_{21} & 2d_{22} & d_{23} \\ d_{31} & d_{32} & 2d_{33} \end{bmatrix}$$

with d_{11} = ^{1}u/^{1}x

d_{22} = ^{1}v/^{1}y

d_{33} = ^{1}w/^{1}z

d_{12} = ^{1}u/^{1}y + ^{1}v/^{1}x = d_{21}

d_{13} = ^{1}u/^{1}z + ^{1}w/^{1}x = d_{31}

d_{23} = ^{1}v/^{1}z + ^{1}w/^{1}y = d_{32}

given the strain tensor e_{ij} defined by $e_{ij} = \frac{1}{2}d_{ij}$

Force *Force*(2D part(s), pressure)

Computes a vector variable whose value is the force F defined as:

$$F = pA$$

where p = pressure (a scalar variable)
A = unit area

Note: The force acts in the surface normal direction.

Force 1D *Force1D*(1D planar part(s), pressure)

Computes a vector variable whose value is the force F defined as:

$$F = pL$$

where p = pressure (a scalar variable)
L = unit length times 1

Note: The force acts in the part's normal direction (in plane).

Gradient *Grad* (any part(s), scalar or vector)

Computes a vector variable whose value is the gradient $GRAD_f$ defined as:

$$GRAD_f = \frac{\partial f}{\partial x}\hat{i} + \frac{\partial f}{\partial y}\hat{j} + \frac{\partial f}{\partial z}\hat{k}$$

where f = any scalar variable (or the magnitude of the specified vector)
x, y, z = coordinate directions
i, j, k = unit vectors in coordinate directions

Gradient Approximation *GradApprox* (any part(s), scalar or vector)

Same as Gradient, except all elements are first subdivided into triangles (for 2D) or tetrahedrons (for 3D) and a closed-form solution is done on the subdivided element's nodal values (only applicable for per node variables). This is basically a quicker, linear approximation of the regular gradient.

Gradient Tensor *GradTensor* (2D or 3D part(s), vector)

Computes a tensor variable whose value is the gradient $GRAD_F$ defined as:

$$GRAD_F = \frac{\partial F}{\partial x}\hat{i} + \frac{\partial F}{\partial y}\hat{j} + \frac{\partial F}{\partial z}\hat{k}$$

where F = any vector variable
x, y, z = coordinate directions
i, j, k = unit vectors in coordinate directions

Gradient Tensor Approximation *GradTensorApprox* (any part(s), vector)

Same as Gradient Tensor, except all elements are first subdivided into triangles (for 2D) or tetrahedrons (for 3D) and a closed-form solution is done on the subdivided element's nodal values (only applicable for per node variables). This is basically a quicker, linear approximation of the regular gradient tensor.

Helicity:

Helicity Density *HelicityDensity*(any part(s), velocity)

Computes a scalar variable H_d whose value is:

$$H_d = V \bullet \Omega$$

where: V = Velocity
Ω = Vorticity

Relative **Helicity** *HelicityRelative*(any part(s), velocity)

Computes a scalar variable H_r whose value is:

$$H_r = cos\ \phi = \frac{V \bullet \Omega}{|V||\Omega|}$$

where: ϕ = the angle between the velocity vector and the vorticity vector.

Filtered Relative **Helicity** *HelicityRelFilter*(any part(s), velocity, freestream velocity magnitude).

Computes a scalar variable H_{rf} whose value is:

$$H_{rf} = H_r \quad \text{, if } |H_d| \geq filter$$

or $$H_{rf} = 0 \quad \text{, if } |H_d| < filter$$

where H_r = relative helicity (as described above)

H_d = helicity density (as described above)

$filter = 0.1(V_\infty)^2$

Iblanking Values *IblankingValues* (any structured iblanked part(s))

Computes a scalar variable whose value is the iblanking flag of selected parts.

Integrals:

Line **Integral** *IntegralLine* (1D part(s), scalar or (vector component or magnitude))

Computes a constant variable whose value is the integral of the input variable over the length of the specified 1D part(s).

Surface **Integral** *IntegralSurface* (2D part(s), scalar or (vector component or magnitude))

Computes a constant variable whose value is the integral of the input variable over the surface of the specified 2D part(s).

Volume **Integral** *IntegralVolume* (3D part(s), scalar or (vector component or magnitude))

Computes a constant variable whose value is the integral of the input variable over the volume of the specified 3D part(s).

Length *Length* (any 1D part(s))

Computes a constant variable whose value is the length of selected parts. While any part can be specified, it will only return a nonzero length if the part has 1D elements.

Line Integral See Line Integral under **Integrals**.

Mach Number *Mach* (any part(s), density, total energy, velocity, ratio of specific heats)

Computes a scalar variable whose value is the Mach number M defined as:

$$M = \frac{u}{\sqrt{\frac{\gamma p}{\rho}}} = \frac{u}{c}$$

where m = momentum
ρ = density
u = speed, computed from velocity input.
γ = ratio of specific heats (1.4 for air)
p = pressure (see *Pressure* below)
c = speed of sound

See Total Energy in this section for a description.

MakeScalElem *Make Scalar At Elements* (any part(s), constant)

Assigns the specified constant value to each element, making a scalar variable.

MakeScalNodes *Make Scalar At Nodes* (any part(s), constant)

Assigns the specified constant value to each node, making a scalar variable.

Make Vector *MakeVect* (any part(s), scalar, scalar, scalar or zero)

Computes a vector variable formed from scalar variables. First scalar becomes the X component of the vector, second scalar becomes the Y component, and the third scalar becomes the Z component. A zero can be specified for the third scalar, creating a 2D vector.

Mass-Flux Average *MassFluxAvg* (any part(s), scalar, velocity, density)

Computes a constant variable whose value is the mass flux average b_{avg} defined as:

$$b\mathrm{avg} = \frac{\int_A \rho b(V \bullet N)dA}{\int_A \rho(V \bullet N)dA} = \frac{MassFluxOfScalar}{MassFlux} = \frac{Flow(plist, b\rho V)}{Flow(plist, \rho V)}$$

where b = any scalar variable, i.e. pressure, mach, a vector component, etc.
ρ = density (constant or scalar) variable
V = velocity (vector) variable
dA = area of some 2D domain
N = unit vector normal to dA

Max *Max* (any part(s), scalar or vector, component)

Computes a constant variable whose value is the maximum value of the scalar (or vector component) in the parts selected. The component is not requested if a scalar is selected.

Min *Min* (any part(s), scalar or vector, component)

Computes a constant variable whose value is the minimum value of the scalar (or vector component) in the parts selected.

Moment *MomentBasedOnCurrentCursorToolLocation* (any part(s), vector, component).

Computes a constant variable (the moment about the cursor tool location) whose value is the *x*, *y*, or *z* component of Moment M.

$$M_x = \Sigma(F_y d_z - F_z d_y)$$

$$M_y = \Sigma(F_z d_x - F_x d_z)$$

$$M_z = \Sigma(F_x d_y - F_y d_x)$$

where F_i = force vector component in direction *i* of vector *F*(x,y,z)

= (Fx,Fy,Fz)

d_i = signed moment arm (the perpendicular distance from the line of action of the vector component F_i to the moment axis (which is the current cursor tool position)).

MomentVector *MomentVector* (any part(s), vector, component).

Computes a vector variable (the moment is computed about each point of the selected parts) whose value is the x, y, or z component of Moment M.

$$M_x = \Sigma(F_y d_z - F_z d_y)$$

$$M_y = \Sigma(F_z d_x - F_x d_z)$$

$$M_z = \Sigma(F_x d_y - F_y d_x)$$

where F_i = force vector component in direction i of vector F(x,y,z)
= (Fx,Fy,Fz)
d_i = signed moment arm (the perpendicular distance from the line of action of the vector component F_i to the moment axis (model point position)).

Momentum *Momentum*(any part(s), velocity, density).

Computes a vector variable m, which is:

$$m = \rho V$$

where ρ = density
V = velocity

Node to Element *NodeToElem* (any part(s), node based variable).

Averages a node based variable to produce an element based variable.

Normal *Normal* (2D part(s) or 1D planar part(s))

Computes a vector variable which is the normal to the surface at each node for 2D parts, or for 1D planar parts - lies normal to the 1D elements in the plane of the part.

Normal Constraints *NormC* (2D or 3D part(s), pressure, velocity, viscosity)

Computes a constant variable whose value is the Normal Constraints NC defined as:

$$NC = \int_S \left(-p + \mu \frac{\partial V}{\partial n}\hat{n}\right) dS$$

where p = pressure
V = velocity
μ = dynamic viscosity
n = direction of normal
S = border of a 2D or 3D domain

Normalize Vector *NormVect* (any part(s), vector)

Computes a vector variable whose value is a unit vector U of the given vector V.

$$U = \frac{V(V_x, V_y, V_z)}{\|V\|}$$

where: V = vector variable field

$$\|V\| = \sqrt{V_x^2 + V_y^2 + V_z^2}$$

Offset Variable *OffsetVar*(2D or 3D part(s))

Computes a scalar (or vector) variable defined as the offset value into the field of that variable that exists in the normal direction from the boundary of the part.

Pressure

Pres (any part(s), density, total energy, velocity, ratio of specific heats)

Computes a scalar variable whose value is the pressure p defined as:

$$p = (\gamma - 1)\rho\left(\frac{E}{\rho} - \frac{1}{2}V^2\right)$$

where: m = momentum
E = internal energy
ρ = density
V = velocity = m/ρ
γ = ratio of specific heats (1.4 for air)

Pressure Coefficient

PresCoef (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, freestream velocity magnitude)

Computes a scalar variable which is Pressure Coefficient C_P defined as:

$$C_P = \frac{p - p_i}{\frac{\rho_i V_i^2}{2}}$$

where: p = pressure
p_i = freestream pressure
ρ_i = freestream density
V_i = freestream velocity magnitude

Dynamic **Pressure**

PresDynam (any part(s), density, velocity)

Computes a scalar variable which is Dynamic Pressure q defined as:

$$q = \frac{\rho V^2}{2}$$

where: ρ = density
V = velocity magnitude

See also: Kinetic Energy

Normalized **Pressure**

PresNorm (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound)

Computes a scalar variable which is Normalized Pressure p_n defined as:

$$p_n = p/p_i$$

where: p_i = freestream pressure = $1/\gamma$
γ = ratio of specific heats
p = pressure

Log of Normalized **Pressure**

PresLogNorm (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound)

Computes a scalar variable which is the natural log of Normalized Pressure defined as: $lnp_n = ln(p/p_i)$

where: p_i = freestream pressure = $1/\gamma$
γ =ratio of specific heats
p = pressure

Stagnation **Pressure**

PresStag (any part(s), density, total energy, velocity, ratio of specific heats)
Computes a scalar variable which is the Stagnation Pressure p_o defined as:

$$p_o = p\left(1+\left(\frac{\gamma-1}{2}\right)M^2\right)^{(\gamma/(\gamma-1))}$$

where: p = pressure
γ = ratio of specific heats
M = mach number

Normalized Stagnation **Pressure**

PresNormStag (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, freestream velocity magnitude)
Computes a scalar variable which is Normalized Stagnation Pressure p_{on}

defined as: $p_{on} = p_o/p_{oi}$

where: p_o = stagnation pressure
p_{oi} = freestream stagnation pressure

Stagnation **Pressure** Coefficient

PresStagCoef (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, freestream velocity magnitude)
Computes a scalar variable which is Stagnation Pressure Coefficient C_{p_o}

defined as: $C_{p_o} = (p_o - p_i)/\left(\frac{\rho_i V^2}{2}\right)$

where: p_o = stagnation pressure
p_i = freestream pressure = $1/\gamma$
γ = ratio of specific heats
ρ_i = freestream density
V = velocity magnitude

Pitot **Pressure**

PresPitot (any part(s), density, total energy, velocity, ratio of specific heats)
Computes a scalar variable which is Pitot Pressure p_p defined as:

$$p_p = sp$$

$$s = \frac{\frac{\gamma+1}{2}\left(\frac{V^2}{\gamma(\gamma-1)\left(\frac{E}{\rho}-\frac{V^2}{2}\right)}\right)^{(\gamma/(\gamma-1))}}{\left(\left(\left(\frac{2\gamma}{\gamma+1}\right)\left(\frac{V^2}{\gamma(\gamma-1)\left(\frac{E}{\rho}-\frac{V^2}{2}\right)}\right)\right)-\left(\frac{\gamma-1}{\gamma+1}\right)\right)^{(1/(\gamma-1))}}$$

where γ = ratio of specific heats
E = total energy per unit volume
ρ = density
V = velocity magnitude
p = pressure

Note: *For mach numbers less than 1.0, the Pitot Pressure is the same as the Stagnation Pressure. For mach numbers greater than or equal to 1.0, the Pitot Pressure is equivalent to the Stagnation Pressure behind a normal shock.*

Pitot **Pressure** Ratio

PresPitotRatio (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound)

Computes a scalar variable which is Pitot Pressure Ratio p_{pr} defined as:

$$p_{pr} = s(\gamma - 1)\left(E - \frac{\rho V^2}{2}\right)$$

where s = (defined above in Pitot Pressure)
γ = ratio of specific heats
E = total energy per unit volume
ρ = density
V = velocity magnitude

Total **Pressure**

PresT (any part(s), pressure, velocity, density)

Computes a scalar variable whose value is the total pressure p_t defined as:

$$p_t = p + \rho\left(\frac{V^2}{2}\right)$$

where ρ = density
V = velocity
p = pressure

Rectangular To Cylindrical Vector

RectToCyl (any part(s), vector)

Produces a vector variable with cylindrical components according to frame 0. (Intended for calculation purposes)

x = radial component, y = tangential component, z = z component

Shock Plot3d

ShockPlot3d(2D or 3D part(s), density, total energy, velocity, ratio of specific heats).

computes a scalar variable *ShockPlot3d*, whose value is:

$$ShockPlot3d = \frac{V}{c} \bullet \frac{grad(p)}{|grad(p)|}$$

where V = velocity
c = speed of sound
p = pressure
$grad(p)$ = gradient of pressure

Spatial Mean

SpaMean (any part(s), scalar or vector, component)

Computes a constant variable whose value is the mean value of a scalar (or vector component) at the current time. This value can change with time. The component is not requested if a scalar variable is used.

Speed

Speed (any part(s), velocity)

Computes a scalar variable whose value is the Speed defined as:

$$Speed = \sqrt{u^2 + v^2 + w^2}$$

where: u,v,w = velocity components in the x,y,z directions.

Sonic **Speed**

SonicSpeed(any part(s), density, total energy, velocity, ratio of specific heats).

Computes a scalar variable c, whose value is:

$$c = \sqrt{\frac{\gamma p}{\rho}}$$

where γ = ratio of specific heats
ρ = density
p = pressure

Stream Function *Stream Function* (any 2D part(s), velocity, density)

Computes a scalar variable whose value is the Stream Function Ψ defined as:

$$\psi = -vdx + udy$$

where: u,v = velocity components in X, Y directions

Surface Integral See Surface Integral under **Integrals**.

Computes a constant variable whose value is the integral of the input variable over the surface of the specified 2D part(s).

Swirl *Swirl*(any part(s), density, velocity).

Computes a scalar variable *Swirl*, whose value is:

$$Swirl = \frac{\Omega \bullet V}{\rho V^2}$$

where: Ω = vorticity

ρ = density

V = velocity

Temperature *Temperature* (any part(s), density, total energy, velocity, ratio of specific heats, gas constant)

Computes a scalar variable whose value is the temperature T defined as:

$$T = \frac{\gamma - 1}{R}\left(\frac{E}{\rho} - \frac{1}{2}V^2\right)$$

where: m = momentum

E = total energy per unit volume

ρ = density

V = velocity = m/ρ

γ = ratio of specific heats (1.4 for air)

R = gas constant

Normalized **Temperature** *TemperNorm* (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, gas constant)

Computes a scalar variable which is Normalized Temperature T_n defined as:

$$T_n = \frac{T}{T_i}$$

where: T = temperature

T_i = freestream temperature

Log of Normalized **Temperature** *TemperLogNorm* (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, gas constant)

Computes a scalar variable which is the natural log of Normalized Temperature

defined as: $ln T_n = ln(T/T_i)$

where: T = temperature

T_i = freestream temperature

Stagnation **Temperature** *TemperStag* (any part(s), density, total energy, velocity, ratio of specific heats, gas constant)

Computes a scalar variable which is the Stagnation Pressure T_o

defined as: $$T_o = T\left(1 + \left(\frac{\gamma - 1}{2}\right)M^2\right)$$

where: T = temperature

γ = ratio of specific heats

M = mach number

Normalized Stagnation **Temperature**

TemperNormStag (any part(s), density, total energy, velocity, ratio of specific heats, freestream density, freestream speed of sound, freestream velocity magnitude, gas constant)

Computes a scalar variable which is Normalized Stagnation Temperature T_{on}

defined as: $T_{on} = T_o / T_{oi}$

where: T_o = stagnation temperature

T_{oi} = freestream stagnation temperature

Temporal Mean

TempMean (any part(s), scalar or vector, time1, time2)

Computes a scalar or vector variable, depending on which type was selected, whose value is the mean value at each node of a scalar or vector variable over the interval from time1 to time2. Thus, the resultant scalar or vector is independent of time.

Tensor:

Tensor Component

Tensor-Component(any part(s), tensor, row, col)

Creates a scalar variable which is the specified row and column of a tensor variable.

S = Tij

i = given row (1 to 3)
j = given column (1 to 3)

Tensor Determinate

Tensor-Determinant(any part(s), Tensor or 3 Principals or 6 Tensor Components)

Computes the determinate of a tensor variable. The tensor may be specified as either a tensor, three principal values or six tensor components. If the three tensor components are given they must be given in the order:

T11, T22, T33, T12, T13,T23.

Tensor Eigenvalue

Tensor-Eigenvalue(any part(s), tensor, number)

Computes the number (1-3) eigenvalue of the given tensor. The first eigenvalue is always the largest, while the third eigenvalue is always the smallest.

Tensor Eigenvector

Tensor-Eigenvector(any part(s), tensor, number)

Computes the number (1-3) eigenvector of the given tensor.

Tensor Make

Tensor-Make(any part(s), T11, T22, T33, T12, T13, T23)

Create a tensor from six scalars.

Tensor Tresca

Tensor-Tresca(any part(s), Tensor or 3 Principals or 6 Tensor Components)

Computes Tresca stress/strain from a tensor variable. The tensor may be specified as either a tensor, three principal values or six tensor components. If the three tensor components are given they must be given in the order:

T11, T22, T33, T12, T13, T23.

Tensor Von Mises

TensorVonMises(any part(s), Tensor or 3 Principals or 6 Tensor Components)

Computes Von Mises stress/strain from a tensor variable. The tensor may be specified as either a tensor, three principal values or six tensor components. If the three tensor components are given they must be given in the order:

T11, T22, T33, T12, T13, T23.

Velocity

Velo (any part(s), momentum, density)

Computes a vector variable whose value is the velocity V defined as:

$$V = \frac{m}{\rho}$$

where ρ = density

m = momentum

Volume

Vol (3D part(s))

Computes a constant variable whose value is the volume of 3D parts.

Volume Integral See Volume Integral under **Integrals**.

Vorticity *Vort* (any 2D or 3D part(s), velocity)

Computes a vector variable with components ζ_x , ζ_y , ζ_z defined as:

$$\zeta_x = \frac{\partial w}{\partial y} - \frac{\partial v}{\partial z} \qquad \zeta_y = \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} \qquad \zeta_z = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$$

where u,v,w = velocity components in the X, Y, Z directions.

Math Math functions use the syntax: function (value *or expression*). All angle arguments are in radians. For most functions the value can be either a constant, scalar, or vector and the result of the function will be of corresponding type. When you select a math function from the list, the function name and the opening "(" appears in the Working Expression for you. However, after defining the argument(s) for the function, you have to manually provide any commas needed and a closing ")". The Math functions include:

Routines which accept a single argument of type constant, scalar, or vector and produce the corresponding type of result: (function works on each component of a vector)	
ABS(constant) **absolute value** = constant *or* (scalar) scalar *or* (vector) vector	***LOG***(constant) **ln** = constant *or* (scalar) scalar *or* (vector) vector
ACOS(radian constant) **arccosine** = constant *or* (radian scalar) scalar *or* (radian vector) vector	***LOG10***(constant) **log$_{10}$** = constant *or* (scalar) scalar *or* (vector) vector
ASIN(radian constant) **arcsine** = constant *or* (radian scalar) scalar *or* (radian vector) vector	***LT***(constant) **less than** = constant or (scalar) scalar or (vector) vector
ATAN(radian constant) **arctangent** = constant *or* (radian scalar) scalar *or* (radian vector) vector	***RND***(constant) **round to nearest** = constant or (scalar) scalar or (vector) vector
COS(radian constant) **cosine** = constant *or* (radian scalar) scalar *or* (radian vector) vector	***SIN***(radian constant) **sine** = constant *or* (radian scalar) scalar *or* (radian vector) vector
EXP(constant) **e$^{\text{value}}$** = constant *or* (scalar) scalar *or* (vector) vector	***SQRT***(constant) **square root** = constant *or* (scalar) scalar *or* (vector) vector
GT(constant) **greater than** = constant or (scalar) scalar or (vector) vector	***TAN***(radian constant) **tangent** = constant *or* (radian scalar) scalar *or* (radian vector) vector

Routines which accept one or more vectors:	
CROSS(vector, vector) **cross product** = vector	***RMS***(vector) **root-mean-square** (magnitude) = scalar
DOT(vector, vector) **dot product** = scalar	

Active Variables Selection list of all variables which are active and therefore available for use in Expressions. You activate variables in the Feature Detail Editor Variables List.

Feedback This area displays interactive guidance when you select a General function, including detailed instructions concerning the function's arguments.

Okay Click this button when so prompted by the Feedback instructions. It basically signals the completion of various intermediate tasks for general functions.

Calculator

This on-screen calculator can usually be used in place of typing on your keyboard.

Button	Function
0 to 9	number digits
.	decimal
e	e for exponential notation
+	plus operator
–	minus operator
*	multiplication operator
/	division operator
^	exponentiation operator
PI	value for π
(	opening parentheses. For function arguments and general grouping
)	closing parentheses. For function arguments and general grouping
[	opening brackets. For components and node/element numbers
]	closing brackets. For components and node/element numbers
[X]	X component
[Y]	Y component
[Z]	Z component

(see How To Create New Variables)